This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. (Currently Amended) A system for index key normalization comprising a processor adapted for:

(a) selecting a column of an index key;

(b) generating a marker corresponding to the selected column;

(c) generating a normalized column value corresponding to the selected column ,wherein the processor generates the normalized column value by determining the type of the column value, and applying a type specific mapping function to the column value; and

(d) appending the marker and the normalized column value pair to a previously generated marker and normalized column value pair if any.

2. (Original) The system of claim 1, wherein the processor repeats steps (a) – (d) for each column in the index key.

3. (Original) The system of claim 1, wherein the processor, after generating the marker, determines if a column value is null, determines if a column value is of type bit, and determines if the column is sorted in ascending or descending order.

4. (Original) The system of claim 3, wherein the processor modifies the marker to indicate if the column value is null, is of type bit, and if the column is sorted in ascending or descending order.

5. (Cancelled)

6. (Currently Amended) The system of claim 1 ~~5~~, wherein the processor compares the normalized column value to other normalized column values independent of the original column type.

7. (Original) The system of claim 1, wherein the processor standardizes the size of the marker and the normalized column value pair before appending the marker and the normalized column value pair to the previously generated marker and normalized column value pair if any.

8. (Original) The system of claim 7, wherein the processor standardizes the size of the marker and the normalized column value pair by comparing the marker and normalized column value pair to a predetermined maximum size and computing a checksum using the pair if the pair is greater than the predetermined maximum size; truncating the marker and normalized column value pair by removing bits from the end of the pair in excess of the predetermined maximum size; and replacing the end bytes of the truncated pair with the computed checksum.

9. (Previously Presented) A system for index key column unnormalization of a normalized index key comprising a processor adapted for:

determining if the type of a selected column value can be unnormalized; and

if so, determining if the selected column was truncated; and

generating the unnormalized column value if the selected column was not truncated.

10. (Original) The system of claim 9, wherein the processor moves through the normalized index key until the selected column is reached, by:

(a) determining if the current column is a fixed size or variable size type; and

(b) if the current column is a fixed size type, moving forward in the normalized index key a number of bytes equal to the size of the fixed size type, or if the current column is a variable size type, moving forward in the normalized index key a number of bytes equal to the length of the column, determined by examining each byte of the current column until the end of the column is reached.

11. (Original) The system of claim 10, wherein the processor repeats steps (a) and (b) for each column in the normalized index key until the selected column is reached.

12. (Original) The system of claim 9, wherein the processor determines if the selected column was truncated by determining if the selected column is a fixed size or variable size type; and, if the selected column is a fixed size type, determining if the size of the previous columns in the index key plus the size of the selected column is greater than a predetermined maximum size, or, if the selected column is a variable size type, examining each byte of the selected column until the end of the column is reached or the number of bytes examined plus the size of the previous columns in the index key exceed a predetermined maximum size.

13. (Currently Amended) A computer-readable medium having stored thereon computer-executable instructions for performing a method for index key normalization comprising:

(a) selecting a column of an index key;

(b) generating a marker corresponding to the selected column;

(c) generating a normalized column value corresponding to the selected column, wherein generating the normalized column value comprises determining the type of the column value, and applying a type specific mapping function to the column value; and

(d) appending the marker and the normalized column value pair to a previously generated marker and normalized column value pair if any.

14. (Original) The computer-readable medium of claim 13, further comprising computer-executable instructions for repeating steps (a) – (d) for each column in the index key.

15. (Original) The computer-readable medium of claim 13, further comprising computer-executable instructions for, after generating the marker:

determining if a column value is null;
determining if a column value is of type bit; and
determining if the column is sorted in ascending or descending order.

16. (Original) The computer-readable medium of claim 15, further comprising computer-executable instructions for modifying the marker to indicate if the column value is null, is of type bit, and if the column is sorted in ascending or descending order.

17. (Cancelled)

18. (Currently Amended) The computer-readable medium of claim 13 ~~17~~, further comprising computer-executable instructions for comparing the normalized column value to other normalized column values independent of the original column type.

19. (Original) The computer-readable medium of claim 13, further comprising computer-executable instructions for standardizing the size of the marker and the normalized column value pair before appending the marker and the normalized column value pair to the previously generated marker and normalized column value pair if any.

20. (Original) The computer-readable medium of claim 19, wherein standardizing the size of the marker and the normalized column value pair comprises:

comparing the marker and normalized column value pair to a predetermined maximum size and computing a checksum using the pair if the pair is greater than the predetermined maximum size;

truncating the marker and normalized column value pair by removing bits from the end of the pair in excess of the predetermined maximum size; and

replacing the end bytes of the truncated pair with the computed checksum.

21. (Previously Presented) A computer-readable medium having stored thereon computer-executable instructions for performing a method for index key column unnormalization of a normalized index key comprising:

determining if the type of a selected column value can be unnormalized; and

if so, determining if the selected column was truncated; and

generating the unnormalized column value if the selected column was not truncated.

22. (Original) The computer-readable medium of claim 21, further comprising computer-executable instructions for moving through the normalized index key until the selected column is reached, by:

(a) determining if the current column is a fixed size or variable size type; and

(b) if the current column is a fixed size type, moving forward in the normalized index key a number of bytes equal to the size of the fixed size type, or if the current column is a variable size type, moving forward in the normalized index key a number of bytes equal to the length of the column, determined by examining each byte of the current column until the end of the column is reached.

23. (Original) The computer-readable medium of claim 22, further comprising computer-executable instructions for repeating steps (a) and (b) for each column in the normalized index key until the selected column is reached.

24. (Original) The computer-readable medium of claim 21, wherein determining if the selected column was truncated comprises:

determining if the selected column is a fixed size or variable size type; and

if the selected column is a fixed size type, determining if the size of the previous columns in the index key plus the size of the selected column is greater than a predetermined maximum size, or, if the selected column is a variable size type, examining each byte of the selected column until the end of the column is reached or the number of bytes examined plus the size of the previous columns in the index key exceed a predetermined maximum size.

25. (Currently Amended) A method for index key normalization comprising:

(a) selecting a column of an index key;

(b) generating a marker corresponding to the selected column;

(c) generating a normalized column value corresponding to the selected column, wherein generating the normalized column value comprises determining the type of the column value, and applying a type specific mapping function to the column value; and

(d) appending the marker and the normalized column value pair to a previously generated marker and normalized column value pair if any.

26. (Original) The method of claim 25, further comprising repeating steps (a) – (d) for each column in the index key.

27. (Original) The method of claim 25, further comprising, after generating the marker:

determining if a column value is null;

determining if a column value is of type bit; and

determining if the column is sorted in ascending or descending order.

28. (Original) The method of claim 27, further comprising modifying the marker to indicate if the column value is null, is of type bit, and if the column is sorted in ascending or descending order.

29. (Cancelled)

30. (Currently Amended) The method of claim 25 ~~29~~, further comprising comparing the normalized column value to other normalized column values independent of the original column type.

31. (Original) The method of claim 25, further comprising standardizing the size of the marker and the normalized column value pair before appending the marker and the normalized column value pair to the previously generated marker and normalized column value pair if any.

32. (Original) The method of claim 31, wherein standardizing the size of the marker and the normalized column value pair comprises:

comparing the marker and normalized column value pair to a predetermined maximum size and computing a checksum using the pair if the pair is greater than the predetermined maximum size;

truncating the marker and normalized column value pair by removing bits from the end of the pair in excess of the predetermined maximum size; and

replacing the end bytes of the truncated pair with the computed checksum.

33. (Previously Presented) A method for index key column unnormalization of a normalized index key comprising:

determining if the type of a selected column value can be unnormalized; and

if so, determining if the selected column was truncated; and

generating the unnormalized column value if the selected column was not truncated.

34. (Original) The method of claim 33, further comprising moving through the normalized index key until the selected column is reached, by:

(a) determining if the current column is a fixed size or variable size type; and

(b) if the current column is a fixed size type, moving forward in the normalized index key a number of bytes equal to the size of the fixed size type, or if the current column is a variable size type, moving forward in the normalized index key a number of bytes equal to the length of the column, determined by examining each byte of the current column until the end of the column is reached.

35. (Original) The method of claim 34, further comprising repeating steps (a) and (b) for each column in the normalized index key until the selected column is reached.

36. (Original) The method of claim 33, wherein determining if the selected column was truncated comprises:

determining if the selected column is a fixed size or variable size type; and

if the selected column is a fixed size type, determining if the size of the previous columns in the index key plus the size of the selected column is greater than a predetermined maximum size, or, if the selected column is a variable size type, examining each byte of the selected column until the end of the column is reached or the number of bytes examined plus the size of the previous columns in the index key exceed a predetermined maximum size.